MAIN SOURCE CODE

Title : Curling Counter and Calories Burned Estimated

Name : Ukhem Fahmi Thoriqul Haq

Import Library

In [ ]:

```python
from tkinter import messagebox
from googleapiclient import discovery
from google.oauth2 import service_account

from tkinter import *
from tkinter import ttk

from datetime import datetime, timedelta
from threading import Timer

import numpy as np
from PIL import Image, ImageTk
import mediapipe as mp
import cv2
import requests
```

Membuat Constant Variable

In [ ]:

```python
# CONSTANT TKINTER
PINK = "#e2979c"
RED = "#e7305b"
GREEN = "#9bdeac"
YELLOW = "#f7f5dd"
FONT_NAME = "Courier"

# CONSTANT CV2 AND MEDIAPIPE
FONT = cv2.FONT_HERSHEY_SIMPLEX
TANGAN_KANAN = [12, 14, 16]
TANGAN_KIRI = [11, 13, 15]

# CONSTANT NUTRITIONIX-API
APP_ID = "*some-ids"
API_KEY = "*some-keys"
EXERCISE_ENDPOINT = "https://trackapi.nutritionix.com/v2/natural/exercise"

# CONSTANT GOOGLE SHEETS APIs
SCOPES = ['https://www.googleapis.com/auth/spreadsheets']
SERVICE_ACCOUNT_FILE = 'gspread-config/service_biomekanika.json'
CREDENTIALS = service_account.Credentials.from_service_account_file(
    filename=SERVICE_ACCOUNT_FILE, scopes=SCOPES)

SERVICE = discovery.build('sheets', 'v4', credentials=CREDENTIALS)
SPREADSHEET_ID = "*some-spreadsheets-ids"

INPUT_USER = "USER_ENTERED"
```

Membuat Variable, Function dan Object Tkinter

In [ ]:

```python
# VARIABLES
calories = 0
counter = 0
duration = 0
count_down = 0
stage = None

pesan_natural = f"I do curl {count_down} seconds"

nutritionix_headers = {
    "x-app-id": APP_ID,
    "x-app-key": API_KEY
}

# Object TKINTER
window = Tk()
window.title("Workout UMUM")
window.geometry("650x550")
window.rowconfigure(0, weight=1)
window.columnconfigure(0, weight=1)
window.config(padx=0, pady=0, background="white")

page_input = Frame(window)
page_cv2 = Frame(window)

label_input = Label(page_input)
label_cv2 = Label(page_cv2)

for tiap_frame in (page_input, page_cv2):
    tiap_frame.grid(row=0, column=0, sticky="nsew")


def show_frame(frame):
    frame.tkraise()
```

Membuat Frame dan Halaman Utama

In [ ]:

```python
# --------------------------------------------------- FRAME INPUT --------------------------------------------------- #
def kanan():
    global yang_dilatih, cetak_tangan
    yang_dilatih = TANGAN_KANAN
    cetak_tangan = "Kanan"
    detect()


def kiri():
    global yang_dilatih, cetak_tangan
    yang_dilatih = TANGAN_KIRI
    cetak_tangan = "Kiri"
    detect()


# BICEPS CURL IMAGE
curl_image = ImageTk.PhotoImage(Image.open("images/biceps.png").resize((150, 150), Image.Resampling.LANCZOS))
label_curl = Label(page_input, image=curl_image)
label_curl.place(x=250, y=60)

# MY CALORIES TRACKER
label_H1 = Label(page_input, text="Our Workout Trackers", font=(FONT_NAME, 25, "bold"))
label_H1.place(x=123, y=30)

# USERNAME
label_username = Label(page_input, text="USERNAME     :", font=(FONT_NAME, 13, "normal"))
label_username.place(x=190, y=230)
entry_username = Entry(page_input, width=20)
entry_username.place(x=335, y=233)
entry_username.focus()

# GENDER
label_gender = Label(page_input, text="MALE/FEMALE  :", font=(FONT_NAME, 13, "normal"))
label_gender.place(x=190, y=260)
gender_options = ["Male", "Female"]
set_gender = ttk.Combobox(page_input, values=gender_options, width=17)
set_gender.place(x=335, y=263)
# entry_gender = Entry(page_input, width=20)
# entry_gender.place(x=335, y=263)

# WEIGHT
label_weight = Label(page_input, text="WEIGHT - Kg  :", font=(FONT_NAME, 13, "normal"))
label_weight.place(x=190, y=290)
entry_weight = Entry(page_input, width=20)
entry_weight.place(x=335, y=293)

# HEIGHT
label_height = Label(page_input, text="HEIGHT - Cm  :", font=(FONT_NAME, 13, "normal"))
label_height.place(x=190, y=320)
entry_height = Entry(page_input, width=20)
entry_height.place(x=335, y=323)

# AGE
label_age = Label(page_input, text="AGE          :", font=(FONT_NAME, 13, "normal"))
label_age.place(x=190, y=350)
entry_age = Entry(page_input, width=20)
entry_age.place(x=335, y=353)

# LABEL PILIH TANGAN YANG AKAN DILATIH
label_pilih = Label(page_input, text="PILIH TANGAN YANG AKAN DILATIH", font=(FONT_NAME, 12, "bold"))
label_pilih.place(x=175, y=410)

# KIRI BUTTON
button_kiri = Button(page_input, text="KIRI", bg="#1746A2", fg="white", width=7, border=0,
                     font=(FONT_NAME, 12, "bold"), command=kiri)
button_kiri.place(x=200, y=440)
# KANAN BUTTON
button_kanan = Button(page_input, text="KANAN", bg="#1746A2", fg="white", width=7, border=0,
                      font=(FONT_NAME, 12, "bold"), command=kanan)
button_kanan.place(x=380, y=440)
```

Membuat Frame CV2 (OpenCV)

In [ ]:

```python
# ---------------------------------------------------- FRAME CV2 ---------------------------------------------------- #
def reset_counter():
    global counter, calories, uncalled, start_1_menit, count_down
    counter = 0
    calories = 0
    count_down = 0
    uncalled = True
    start_1_menit.cancel()


def save_data():
    global start_1_menit
    start_1_menit.cancel()
    pencatatan_akhir()
    exit()


# RESET BUTTON
button_reset = Button(page_cv2, text="RESET", bg="red", fg="white", width=7, border=0,
                      font=(FONT_NAME, 12, "bold"), command=reset_counter)
button_reset.place(x=180, y=500)
# SAVE BUTTON
button_save = Button(page_cv2, text="SAVE", bg="#5BB318", fg="white", width=7, border=0,
                     font=(FONT_NAME, 12, "bold"), command=save_data)
button_save.place(x=400, y=500)
```

Function Membuka Page-1

In [ ]:

```python
# --------------------------------------------------- BUKA PAGE-1 --------------------------------------------------- #

show_frame(page_input)

# FRAME LOKASI LABEL CV2
label_cv2.place(x=3, y=5)
```

Function Membuka Page-2 - CV2 & MediaPipe

In [ ]:

```python
"""  ------------------------------------------- PAGE-2 & CV2-MEDIAPIPE ------------------------------------------- """

# ----------------------------------------- NUTRITIONIX API REQUEST FOR LAST ----------------------------------------- #

def pencatatan_akhir():
    pesan_natural = f"I do curl {duration} seconds"
    exercise_parameters = {
        "query": pesan_natural,
        "gender": isi_gender,
        "weight_kg": isi_weight,
        "height_cm": isi_height,
        "age": isi_age
    }

    exercise_response = requests.post(url=EXERCISE_ENDPOINT, json=exercise_parameters,
                                      headers=nutritionix_headers)
    # print(exercise_response.text)
    nutritionix_response = exercise_response.json()
    print(nutritionix_response)

    # ambil kalori
    calories = nutritionix_response["exercises"][0]["nf_calories"]
    exercise = nutritionix_response["exercises"][0]["name"].title()
    print(f"exercise: {exercise}, duration: {duration}, calories: {calories}, gender: {isi_gender}, nama: {isi_nama}")

    # ----------------------------------------- KIRIM DATA KE GOOGLE SHEETS ----------------------------------------- #
    now = datetime.today()
    date = now.strftime("%d/%m/%Y")
    time = now.strftime("%H:%M:%S")
    duration_min = round((duration / 60), 1)
    RANGE_POST = "Umum!A1:H1"

    data = [[isi_nama, date, time, exercise, cetak_tangan, counter, duration_min, calories]]

    request = SERVICE.spreadsheets().values().append(spreadsheetId=SPREADSHEET_ID, range=RANGE_POST,
                                                     valueInputOption=INPUT_USER, insertDataOption="INSERT_ROWS",
                                                     body={"values": data})

    response = request.execute()
    print(response)


def satu_menit():
    global count_down, calories, pesan_natural, exercise, start_1_menit
    start_1_menit = Timer(60, satu_menit)
    start_1_menit.start()

    pesan_natural = f"I do curl {count_down} seconds"
    count_down += 60

    exercise_parameters = {
        "query": pesan_natural,
        "gender": isi_gender,
        "weight_kg": isi_weight,
        "height_cm": isi_height,
        "age": isi_age
    }

    exercise_response = requests.post(url=EXERCISE_ENDPOINT, json=exercise_parameters,
                                      headers=nutritionix_headers)
    # print(exercise_response.text)
    nutritionix_response = exercise_response.json()
    print(nutritionix_response)

    # Ambil Kalori Terbakar & Latihan Yang Dilakukan
    calories = nutritionix_response["exercises"][0]["nf_calories"]
    exercise = nutritionix_response["exercises"][0]["name"].title()


def calculate_angle(a, b, c):
    global angle
    a = np.array(a)  # First
    b = np.array(b)  # Mid
    c = np.array(c)  # End

    radians = np.arctan2(c[1] - b[1], c[0] - b[0]) - np.arctan2(a[1] - b[1], a[0] - b[0])
    angle = np.abs(radians * 180.0 / np.pi)

    if angle > 180.0:
        angle = 360 - angle

    return angle


cap = cv2.VideoCapture(1)
mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
pose = mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5)

uncalled = True


# def do_mediapipe():
def detect():
    global isi_gender, isi_weight, isi_height, isi_age, isi_nama
    isi_gender = set_gender.get()
    isi_weight = entry_weight.get()
    isi_height = entry_height.get()
    isi_age = entry_age.get()
    isi_nama = entry_username.get().title()
    if len(isi_gender) == 0 or len(isi_weight) == 0 or len(isi_height) == 0 or len(isi_age) == 0 or len(isi_nama) == 0:
        messagebox.showinfo(title="UPSS",
                            message="NANTI DULU !!!!!"
                                    "\nJangan Kosongkan Data")

    else:
        show_frame(page_cv2)
        global counter, stage, waktu_start, duration, uncalled

        ret, frame = cap.read()
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        image.flags.writeable = False

        # Make detection
        results = pose.process(image)

        # Recolor back to BGR
        image.flags.writeable = True
        # image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

        # --------------------------------------------- TIMER API START --------------------------------------------- #

        try:
            if counter > 0 and uncalled:
                satu_menit()
                uncalled = False
            else:
                pass
        except:
            pass

        # -------------------------------------------- EXTRACT LANDMARKS -------------------------------------------- #

        # Extract Landmarks
        try:
            landmarks = results.pose_landmarks.landmark

            # Get Coordinate
            shoulder = [landmarks[yang_dilatih[0]].x, landmarks[yang_dilatih[0]].y]
            elbow = [landmarks[yang_dilatih[1]].x, landmarks[yang_dilatih[1]].y]
            wrist = [landmarks[yang_dilatih[2]].x, landmarks[yang_dilatih[2]].y]

            # Calculate Angle
            angle = calculate_angle(shoulder, elbow, wrist)

            # Visualize Angle
            cv2.putText(image,
                        str(round(angle, 2)),
                        tuple(np.multiply(elbow, [640, 480]).astype(int)),
                        FONT, 0.5, (255, 255, 255), 2, cv2.LINE_AA)

            waktu_sekarang = datetime.now()

            if counter < 1:
                waktu_start = datetime.now()

            # Curl counter Logic
            if angle > 160:
                stage = "Down"
            if angle < 45 and stage == "Down":
                stage = "Up"
                counter += 1

        except:
            pass

        duration = waktu_sekarang - waktu_start
        duration = duration.seconds

        # ------------------------------------------------- DISPLAY ------------------------------------------------- #
        # Setup status box
        cv2.rectangle(image, (0, 0), (640, 80), (245, 117, 16), -1)

        # Repetisi Data
        cv2.putText(image, "REPETISI", (15, 25),
                    FONT, 0.55, (0, 0, 0), 1, cv2.LINE_AA)
        cv2.putText(image, str(counter), (22, 67),
                    FONT, 1.5, (255, 255, 255), 2, cv2.LINE_AA)

        # Stage Data
        cv2.putText(image, "STAGE", (135, 25),
                    FONT, 0.55, (0, 0, 0), 1, cv2.LINE_AA)
        cv2.putText(image, stage, (120, 65),
                    FONT, 1, (255, 255, 255), 2, cv2.LINE_AA)

        # Start Time
        cv2.putText(image, "TIMER", (270, 25),
                    FONT, 0.55, (0, 0, 0), 1, cv2.LINE_AA)
        cv2.putText(image, str(timedelta(seconds=duration)), (235, 65),
                    FONT, 1, (255, 255, 255), 2, cv2.LINE_AA)

        # Tangan Latih
        cv2.putText(image, "TANGAN LATIH", (380, 25),
                    FONT, 0.55, (0, 0, 0), 1, cv2.LINE_AA)
        cv2.putText(image, cetak_tangan, (390, 65),
                    FONT, 1, (255, 255, 255), 2, cv2.LINE_AA)

        # Kalori
        cv2.putText(image, "KALORI", (540, 25),
                    FONT, 0.55, (0, 0, 0), 1, cv2.LINE_AA)
        cv2.putText(image, str(calories), (540, 65),
                    FONT, 1, (255, 255, 255), 2, cv2.LINE_AA)

        # -------------------------------------------- RENDER DETECTIONS -------------------------------------------- #

        # Render detections
        mp_drawing.draw_landmarks(image, results.pose_landmarks, mp_pose.POSE_CONNECTIONS,
                                  mp_drawing.DrawingSpec(color=(245, 117, 66), thickness=2, circle_radius=2),
                                  mp_drawing.DrawingSpec(color=(66, 117, 245), thickness=2, circle_radius=2),
                                  )

        img = image[:, :640, :]
        imgarr = Image.fromarray(img)
        imgtk = ImageTk.PhotoImage(imgarr)
        label_cv2.imgtk = imgtk
        label_cv2.configure(image=imgtk)
        label_cv2.after(10, detect)


window.mainloop()
```